Schürer, Jürgen

Heft 1

ELEKTRONISCHER KLEINRECHNER

D 4a

VORBEREITUNGSHEFT FÜR D 4 a - LEHRGÄNGE

veb bürotechnik

Schulungszentrum



Inhaltsverzeichnis

0. Einleitung

Das vorliegende Heft hat zur Aufgabe, den Teilnehmern des Lehrganges "Programmierung des elektronischen Kleinrechners D 4a" die notwendigen Vorkenntnisse für eine einheitliche Ausgangsbasis zu vermitteln.

Der behandelte Stoff erhebt keinen Anspruch auf Vollständigkeit, sondern stellt nur eine Einführung dar.

Die Kenntnis des Inhaltes dieses Heftes wird im Lehrgang vorausgesetzt.

1. Allgemeine Charakteristik des D 4a

Der D 4a ist ein programmgesteuerter, elektronischer Digitalrechner und wird in die Kategorie der Kleinrechner eingeordnet.

Kleinrechner zeichnen sich durch einfache Konstruktion, relativ geringe Herstellungskosten und kleine Abmessungen aus.

Im allgemeinen besitzen Kleinrechner nicht die hohen Rechengeschwindigkeiten großer Anlagen, sie haben eine verhältnismäßig kleine Speicherkapazität, es sind nur wenige Grundoperationen verdrahtet. Trotzdem erreichen auch Kleinrechner beachtliche Leistungen.

Die sich widersprechenden Forderungen nach einem niedrigen Preis und einem hohen Leistungsvermögen sind im D 4a durch folgende Maßnahmen weitgehend verwirklicht:

1. Einsatz einer Magnettrommel als dem zur Zeit billigsten Arbeitsspeicher
2. Einfachster Aufbau von Rechen- und Leitwerk
3. Mehrfachnutzung vorhandener Bauelemente
4. Hohe Trommelgeschwindigkeit (kurze Zugriffszeit)
5. Anwendung neuartiger Befehle, die die Möglichkeiten des Trommelspeichers konsequent nutzen.

Digitalrechner sind Ziffernrechner. In ihnen wird eine Anzahl gleicher technischer oder physikalischer Zustände dazu verwendet, Ziffern bzw. Zahlen darzustellen. Das Rechnen läuft in jedem Fall auf das Zählen dieser Zustände hinaus.

Neben hohen Rechengeschwindigkeiten weisen moderne elektronische Rechenautomaten eine sogenannte Programmsteuerung auf. Ein Programm, bestehend aus Einzelanweisungen, wird schrittweise und selbständig vom Rechner abgearbeitet. Rechenvorgänge und logische Operationen werden automatisch ausgeführt. Manuelle Eingriffe erfolgen nur noch bei Störungen oder wenn sie vorgesehen sind.

In der Regel bestehen digitale Rechenautomaten aus folgenden Baugruppen:

- dem Rechenwerk, das die arithmetischen und logischen Operationen ausführt
- dem Speicher, der sowohl das Programm als auch alle benötigten Informationen (Ein- und Ausgabedaten sowie Zwischenergebnisse) aufnimmt
- dem Eingabewerk, das die zu verarbeitenden Informationen übernimmt und weiterleitet (z. B. zum Speicher)
- dem Ausgabewerk, das die geforderten Informationen (z. B. Resultate) ausgibt (in Klarschrift auf Papier oder verschlüsselt auf Datenträger)
- dem Leitwerk, das die einzelnen Baugruppen und ihr Zusammenwirken steuert.

Der D 4a als Kleinrechner besitzt kein spezielles Ein- und Ausgabewerk, sondern benutzt dafür auch für andere Zwecke verwendete Baugruppen (Adressenregister).

Im D 4a, der zur Gruppe der Einadreßrechner gehört, besitzt jeder Befehl (Elementaranweisung) eine Adresse. Diese gibt an, auf welchem Speicherplatz sich die zu verarbeitende Information befindet.

Die Zahlenverarbeitung erfolgt in Serie. Die Ziffern einer Zahl werden schrittweise nacheinander verarbeitet. Im Gegensatz zur Parallelverarbeitung, bei der die Ziffern einer Zahl gleichzeitig nebeneinander (parallel) verarbeitet werden und der erforderliche technische Aufwand relativ hoch ist, ist der Aufwand bei Serienverarbeitung auf ein Minimum beschränkt. Die Serienverarbeitung hat jedoch eine niedrigere Arbeitsgeschwindigkeit zur Folge.

Die Eingabe der zu verarbeitenden Informationen erfolgt in der Regel manuell über eine Tastatur oder maschinell über Datenträger.
Die Ausgabe erfolgt über Schreibmaschine bzw. Drucker oder verschlüsselt über Datenträger.

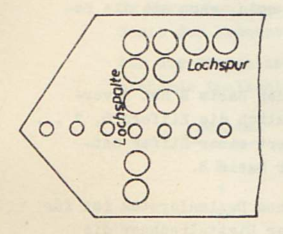
Als einziger Datenträger für den D 4 a kann der Lochstreifen verwendet werden.

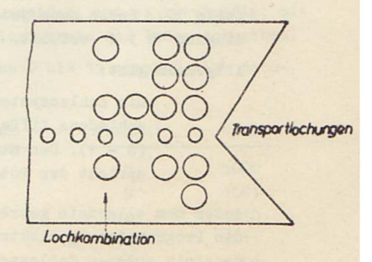
Der Lochstreifen besteht aus einem dünnen Papierstreifen von 17,5 bis 25,4 mm Breite (entsprechend der Spurenzahl).

Im Lochstreifen werden, verschlüsselt in Form von Lochkombinationen, Informationen vielfältiger Art dargestellt. Jede Lochspalte beinhaltet eine Lochkombination, d. h. ein Zeichen. Der Schlüssel (Code) - Anordnung und Anzahl der Lochungen je Spalte - bestimmt den Charakter des Zeichens.

Es gibt 5-, 6-, 7- und 8 spurige Streifen.

Beispiel für 5 spurigen Lochstreifen





2. Zahlensysteme

2.1. Dezimalsystem

Das Dezimalsystem ist ein Stellenwertsystem.
Die Schreibweise als Ziffernfolge stellt eine Abkürzung dar. Die verwendeten Ziffern 0, 1, 2, 3, ... 9 bezeichnen die ganzen Zahlen von Null bis Neun (Ziffernwert), der Wert jeder Ziffer ändert sich aber durch die Stellung der Ziffer innerhalb der Ziffernfolge (Stellenwert = Einer, Zehner, Hunderter, Zehntel usw.).
Die Zahl 617,05 ist eine Abkürzung für die Potenzschreibweise

$$6 \cdot 10^2 + 1 \cdot 10^1 + 7 \cdot 10^0 + 0 \cdot 10^{-1} + 5 \cdot 10^{-2}$$

Einer bestimmten Stelle innerhalb der Zahl ist eine entsprechende Zehnerpotenz zugeordnet. Multipliziert man die einzelnen Ziffern mit der zugehörigen Zehnerpotenz und addiert dann die Produkte, so erhält man den Zahlenwert.

Die Anzahl der zulässigen verschiedenen Ziffern eines Zahlensystems bezeichnet man als Grundzahl oder Basis (hier 10). Basis kann jede Zahl sein, wenn sie die Bedingung $B > 1$ erfüllt.

Allgemein gilt:

> Ein Zahlensystem mit der Basis B hat B verschiedene Ziffern, nämlich die Ziffern 0, 1 ... (B - 1). Der Stellenwert einer Ziffer entspricht der Potenz zur Basis B.

Außer dem allgemein gebräuchlichen Dezimalsystem ist für die Programmierung elektronischer Digitalrechner die Kenntnis anderer Zahlensysteme erforderlich. Diese sind u. a. das Dual- und das Oktalsystem.

2.2. Dualsystem

Das Dualsystem hat die Basis 2.
Es besitzt nur zwei zulässige Ziffern, nämlich 0 und 1 (auch mit O und L bezeichnet).

Der Stellenwert einer Ziffer innerhalb der Dualzahl wird durch eine Potenz von 2 bestimmt.
Die Zahl $1011{,}1_2$ ist eine Abkürzung für die Potenzschreibweise

$$1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1}$$

Multipliziert man den Ziffernwert mit dem Stellenwert und addiert danach die Produkte, so ergibt sich der dezimale Zahlenwert dieser Dualzahl

$$\begin{aligned} 1 \cdot 2^3 &= 8 \\ 0 \cdot 2^2 &= 0 \\ 1 \cdot 2^1 &= 2 \\ 1 \cdot 2^0 &= 1 \\ 1 \cdot 2^{-1} &= \underline{0{,}5} \\ & \quad 11{,}5 \end{aligned}$$

$$1011{,}1_2 = 11{,}5_{10}$$

Um Verwechselungen zu vermeiden, ist es zweckmäßig, die Basis des benutzten Zahlensystems als Index anzugeben.

Für die Dezimalzahlen von 0 bis 15 ergeben sich nachstehende Dualzahlen:

dezimal	dual	dezimal	dual
0	0	8	1000
1	1	9	1001
2	10	10	1010
3	11	11	1011
4	100	12	1100
5	101	13	1101
6	110	14	1110
7	111	15	1111

Im Dualsystem gelten ähnliche Rechenregeln wie im Dezimalsystem.

Addition

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 und Übertrag 1 in die nächsthöhere Stelle

Subtraktion

0 - 0 = 0
1 - 0 = 1
1 - 1 = 0
0 - 1 = 1 und Übertrag -1 in die nächsthöhere Stelle

Multiplikation

0 x 0 = 0
0 x 1 = 0
1 x 0 = 0
1 x 1 = 1

In den nachfolgenden Beispielen für die Anwendung dieser Rechenregeln für das Dualsystem werden evtl. vorkommende Überträge besonders hervorgehoben.

Beispiele für Addition:

	dual	dezimal
1. Beispiel	1 0 0 1	9
	+ 1 0	+ 2
	1 0 1 1	11
2. Beispiel	1 0 1 1 0 1	45
	+ 1 0 1 1 1 0	+ 46
erste vorläufige Summe	0 0 0 0 1 1	91
+ Übertrag	1 1 1	
	1 0 1 1 0 1 1	

Beispiele für Subtraktion

	dual	dezimal
1. Beispiel	1 1 0 1 , 1 1	13,75
	- 1 0 1 0 , 0 1	- 10,25
erste vorläufige Differenz	0 1 1 1 , 1 0	3,50
		====
Übertrag	-1	
	0 0 1 1 , 1 0	
	=============	
2. Beispiel	1 0 1 0 1 , 0 0	21,00
	- 1 0 1 0 0 , 0 1	- 20,25
erste vorläufige Differenz	0 0 0 0 1 , 0 1	0,75
		====
Übertrag	,-1	
zweite vorläufige Differenz	0 0 0 0 1 , 1 1	
Übertrag	-1 ,	
	0 , 1 1	
	=======	

Beispiele für Multiplikation

dual	dezimal
1 0 1 , 1 x 1 1 0 , 1	5,5 · 6,5
1 0 1 1	3 3 0
1 0 1 1	2 7 5
0 0 0 0	3 5,7 5
1 0 1 1	
1 0 0 0 1 1,1 1	
==============	

Dualzahlen werden in elektronischen Digitalrechnern auch logisch verknüpft. Deshalb sollen an dieser Stelle die logischen Operationen kurz erläutert werden.

Konjunktion (die logische Multiplikation, das logische "und"):

Die Konjunktion wird durch das Zeichen $\wedge$ oder & ausgedrückt.

Ziffern mit dem gleichen Stellenwert werden nach folgenden Regeln verknüpft:

$0 \wedge 0 = 0$	Das Ergebnis ist nur
$1 \wedge 0 = 0$	dann gleich 1, wenn
$0 \wedge 1 = 0$	beide verarbeiteten
$1 \wedge 1 = 1$	Ziffern gleich 1 sind.

Beispiel:

$$\begin{array}{r} 1001110 \\ \wedge\ 1100110 \\ \hline 1000110 \end{array}$$

Disjunktion (die logische Addition, das logische "oder")

Die Disjunktion wird durch das Zeichen $\vee$ ausgedrückt.

Ziffern mit dem gleichen Stellenwert werden nach folgenden Regeln verknüpft:

$0 \vee 0 = 0$	Das Ergebnis ist gleich
$0 \vee 1 = 1$	1, wenn eine der beiden
$1 \vee 0 = 1$	verarbeiteten Ziffern
$1 \vee 1 = 1$	gleich 1 ist.

Beispiel:

$$\begin{array}{r} 10010111 \\ \vee\ 10100101 \\ \hline 10110111 \end{array}$$

Negation (Inversion)

Eine Negation wird durch Überstreichen der Zahl ausgedrückt.

Für die Negation gilt folgende Regel:

$\overline{1} = 0$

$\overline{0} = 1$

Beispiel: $\overline{1011001} = 0100110$

2.3. Oktalsystem

Das Oktalsystem ist ein Stellenwertsystem mit der Basis B = 8.

Es treten die Ziffern von 0 bis 7 auf.

Die Stellenwerte der Ziffern innerhalb der Oktalzahl sind durch eine Potenz von 8 bestimmt.

8^0	=	1	8^{-1}	=	0,125
8^1	=	8	8^{-2}	=	0,015625
8^2	=	64	8^{-3}	=	0,001953125
8^3	=	512	8^{-4}	=	0,000244140625
8^4	=	4096			

Die ganzen Dezimalzahlen von 0 bis 20 ergeben folgende Oktalzahlen:

dezimal	oktal	dezimal	oktal
0	0	11	13
1	1	12	14
2	2	13	15
3	3	14	16
4	4	15	17
5	5	16	20
6	6	17	21
7	7	18	22
8	10	19	23
9	11	20	24
10	12		

Die Zahl $132{,}4_8$ ist eine Abkürzung für die Potenzschreibweise:

$$1 \cdot 8^2 + 3 \cdot 8^1 + 2 \cdot 8^0 + 4 \cdot 8^{-1}$$

Als dezimaler Zahlenwert dieser Oktalzahl ergibt sich

$$132{,}4_8 \quad = \quad 90{,}5_{10}$$

Der Vorteil des Oktalsystems besteht darin, daß eine einfache Umrechnung in das Dualsystem möglich ist; außerdem ist die Oktalzahl kürzer als die Dualzahl.

Bei der Addition von Oktalzahlen ist zu beachten, daß ein Übertrag zur nächsthöheren Stelle schon erfolgt, wenn die Teilsumme > 7 wird. Dabei ist es zweckmäßig, zunächst die Ziffern im Dezimalsystem zu addieren. Ist die Summe zweier Ziffern > 7, dann wird von dieser Teilsumme 8 (oder evtl. ein Vielfaches von 8) subtrahiert und die Differenz niedergeschrieben. Die Anzahl der subtrahierten Achten kommt als Übertrag zur nächsthöheren Stelle.

		oktal	dezimal
Beispiel:		132_8	90
	+	367_8	+ 247
Überträge		11	337
			=====
		521_8	
		====	

Für die Subtraktion gilt ähnliches, hier wird aber beim Überschreiten der Ziffer 7 eine 8 (oder evtl. ein Vielfaches von 8) addiert und die letzte Ziffer der Summe niedergeschrieben. Die Anzahl der addierten Achten kommt als Übertrag in die nächsthöhere Stelle.

		oktal	dezimal
Beispiel:		521_8	337
	-	367_8	- 247
Überträge	-	11	90
			=====
		132_8	
		====	

2.4. Umwandlung von einem Zahlensystem in ein anderes

Die Umwandlung von Zahlen in ein anderes Zahlensystem (Konvertierung) wird in der Regel vom Rechner vorgenommen. Für die Programmierung und das Testen von Programmen ist aber oft ein manuelles Umrechnen unumgänglich. Bei der Angabe einiger Umrechnungsverfahren in den folgenden Abschnitten wurde auf Herleitung und Beweis dieser Verfahren verzichtet.

2.4.1. Umwandlung vom Dual- in das Oktalsystem und umgekehrt

Die Umwandlung vom Dual- in das Oktalsystem ist sehr einfach, da die 8 eine Potenz von 2 ist

$$8 = 2^3$$

Deshalb ergibt sich folgende Umrechnungsregel:

Vom Komma ausgehend wird die Dualzahl in Dreiergruppen eingeteilt und diese Dreiergruppen werden durch Oktalziffern ersetzt.

Z. B.: Die Dualzahl $111011101010,000100011_2$ soll in eine Oktalzahl umgewandelt werden.

001 110 111 101 010 , 000 100 011
1 6 7 5 2 , 0 4 3

Die Oktalzahl lautet also:

$$16752,043_8$$

Die Umwandlung einer Oktalzahl in eine Dualzahl erfolgt ähnlich, d. h. die Oktalziffern werden als 3-stellige Dualzahlen geschrieben, die Aneinanderreihung ergibt die vollständige Dualzahl.

2.4.2. Umwandlung von Dezimalzahlen in das Dual- bzw. Oktalsystem

Die Umwandlung ganzer Zahlen aus dem Dezimalsystem in ein Stellenwertsystem mit einer anderen Basis erfolgt durch fortlaufende Division durch die Basis und Festhalten des Divisionsrestes.

Die Division wird solange fortgeführt, bis der letzte Quotient kleiner als die Basis ist. Die umgerechnete Zahl ergibt sich durch Aneinanderfügen der Divisionsreste von unten nach oben, die erste Ziffer ist der letzte Quotient.

Beispiel: Die Dezimalzahl 315 soll als Dual- und als Oktalzahl dargestellt werden.

a) Umwandlung in das Dualsystem

Quotient	Divisionsrest	Erläuterung:
315		315 : 2 = 157 R. 1
157	1	157 : 2 = 78 R. 1
78	1	78 : 2 = 39 R. 0
39	0	usw.
19	1	
9	1	
4	1	
2	0	
1	0	
0	1	

$315_{10} = 100111011_2$ 1 < 2 (letzter Quotient)

b) Umwandlung in das Oktalsystem

Quotient	Divisionsrest	Erläuterung:
315		315 : 8 = 39 R. 3
39	3	39 : 8 = 4 R. 7
4	7	
0	4	

$315_{10} = 473_8$

Es ist also

$$315_{10} = 100\ 111\ 011_2 = 473_8$$

Soll eine echt gebrochene Dezimalzahl in ein anderes Zahlensystem umgewandelt werden, so wird mit der Basis des anderen Zahlensystems fortlaufend multipliziert, wobei

der ganze Teil des jeweiligen Ergebnisses abgespalten wird. Die fortlaufende Multiplikation kann bis zur geforderten Stellenzahl oder bis zum Abbruch fortgesetzt werden. Die Ziffernfolge des ganzen Teils ergibt von oben nach unten gelesen die gesuchte Zahl.

Beispiel: Die Dezimalzahl 0,34375 soll in das Dual- bzw. Oktalsystem umgewandelt werden.

Umwandlg. in das Dualsystem			Umwandlg. in das Oktalsystem		
Produkt	ganzer Teil	Erläuterung	Produkt	ganzer Teil	Erläuterung
0,34375		0,34375·2=0,6875	0,34375		0,34375·8
0,6875	0	0,6875 ·2=1,375	2,75000	2	0,75 ·8
1,375	1	0,375 ·2=0,75	6,0	6	
0,75	0	usw.			
1,5	1				
1,0	1				

$0,34375_{10} = 0,010\ 11_2 = 0,26_8$

Bei diesem Beispiel erfolgte ein Abbruch der Multiplikation, da ein ganzzahliges Produkt entstand, die weiteren Stellen nach dem Komma würden dann nur noch mit 0 besetzt sein.
Bei der Umwandlung von unecht gebrochenen Zahlen erfolgt eine Kombination der beiden o. g. Methoden, d. h. der ganze Teil der Zahl und der echt gebrochene Teil der Zahl werden einzeln berechnet.

$315,34375_{10} = 473,26_8 = 100\ 111\ 011,\ 010\ 11_2$

2.4.3. Umwandlung von Oktalzahlen in Dezimalzahlen

Die Verfahren zur Umwandlung von Oktalzahlen in Dezimalzahlen, die in diesem Abschnitt beschrieben werden, verwenden ganze Oktalzahlen. Eine Oktalzahl mit einem gebrochenen Teil muß zunächst ganzzahlig gemacht werden. Das wird durch Verschieben der Stellenwertigkeit erreicht. In Summenform geschrieben lautet eine Oktalzahl

$$\sum_{i=-n}^{m} x_i \cdot 8^i$$

Zieht man den Faktor 8^{-n} aus der Summe heraus, so ergibt sich

$$\sum_{i=-n}^{m} x_i \cdot 8^i = 8^{-n} \sum_{i=-n}^{m} x_i \cdot 8^{i+n}$$

Beispiel:

$$127{,}36_8 = \sum_{i=-2}^{2} x_i \cdot 8^i = 1 \cdot 8^2 + 2 \cdot 8^1 + 7 \cdot 8^0 + 3 \cdot 8^{-1} + 6 \cdot 8^{-2}$$

$$127{,}36_8 = 8^{-2} \sum_{i=-2}^{2} x_i \cdot 8^{i+2} = 8^{-2} \cdot 12736_8$$

Die Oktalzahl 12736_8 kann nach den beiden in folgendem beschriebenen Verfahren aufgelöst werden; das Ergebnis wird mit 8^{-2} multipliziert, um den dezimalen Zahlenwert für die Oktalzahl $127{,}36_8$ zu erhalten.

Umwandlung mit Hilfe des Horner-Schemas:

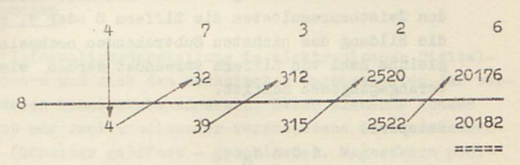
Als Beispiel wird die im Abschnitt 2.4.2. errechnete Oktalzahl $473{,}26_8$ gewählt. Diese Oktalzahl wird zunächst ganzzahlig gemacht:

$$473{,}26_8 = 8^{-2} \cdot 47326_8$$

Die einzelnen Ziffern x_i der ganzzahligen Oktalzahl werden nebeneinander aufgeschrieben. Die erste Ziffer (4) wird nach unten gezogen und mit 8 multipliziert (32). Das Ergebnis wird zur nächsten Ziffer (7) addiert (39) und diese Summe wieder mit 8 multipliziert usw. Das erhaltene Ergebnis nach Addition der letzten Ziffer (20182) ist der dezimale Zahlenwert der ganzzahligen Oktalzahl.

Dieser Wert muß noch mit 8^{-2} multipliziert werden, um den dezimalen Zahlenwert unserer Ausgangszahl zu erhalten.

Horner-Schema:



Korrektur des Kommas:

$$20182 \cdot 8^{-2} = \frac{20182}{64} = 315{,}34375$$

$$473{,}26_8 = 315{,}34375_{10}$$

Eine zweite Methode der Umwandlung ganzer Oktalzahlen in Dezimalzahlen ist die folgend beschriebene:

Die erste Ziffer der Oktalzahl wird mit 2 multipliziert und dieser Wert wird von den ersten beiden Oktalziffern abgezogen. Von diesem Resultat nimmt man die ersten 2 Ziffern, multipliziert sie mit 2 und zieht diesen Wert von den ersten 3 Ziffern des letzten Resultates ab. Von diesem neuen Resultat nimmt man die ersten 3 Ziffern usw. Das Verfahren wird solange fortgesetzt, bis eine erneute Subtraktion eine negative Zahl ergeben würde.

Beispiel:

```
  7 7 7 7₈
- 1 4
  -------
  6 3 7 7
- 1 2 6
  -------
  5 1 1 7
- 1 0 2 2
  -------
  4 0 9 5
  =======
```

$$7777_8 = 4095_{10}$$

Für dieses Verfahren gilt eine Ausnahme. Erscheinen in den Zwischenresultaten die Ziffern 8 oder 9, so muß für die Bildung des nächsten Subtrahenden nochmals die gleiche Zahl von Ziffern verwendet werden wie beim vorangegangenen Schritt.

Beispiel:

```
              1 0 6 4₈
          -     2
---->         8 6 4
          -   1 6       (nicht 172 !)
              7 0 4
          -   1 4 0
              5 6 4
              =====
```

$1064_8 = 564_{10}$

=========================

3. Informationsdarstellung in elektronischen Digitalrechenautomaten

Die Informationsdarstellung in elektronischen Digitalrechnern muß sich den technischen Gegebenheiten der verwendeten Bauelemente anpassen. Diese Elemente können meist nur zwei voneinander verschiedene Zustände annehmen (Schalter geöffnet - geschlossen, Magnetkern positiv oder negativ gesättigt, Elektronenröhre und Transistor leiten oder leiten nicht). Der Rechenautomat besitzt somit einen dualen Charakter; jedem der beiden Zustände der Bauelemente kann ein Symbol "0" bzw. "L" zugeordnet werden. Die Zahlen 0 und 1 lassen sich also recht einfach darstellen. Größere Zahlen werden durch Kombinationen zweiwertiger Elemente gebildet, im Rechner gespeichert und verknüpft. Die kleinste Speichereinheit, die Auskunft über den Zustand eines zweiwertigen Elementes gibt, bezeichnet man als B i t.
Für größere Zahlen werden mehrere Bits zusammengefaßt zu einem W o r t. Sind die Worte immer gleich lang, so spricht man von konstanter Wortlänge.
Da der Mensch gewohnt ist, im Dezimalsystem zu rechnen, die Maschine aber auf dem Dualsystem aufgebaut ist, muß eine gegenseitig eindeutige Zuordnung zwischen dem Dezimalsystem und den dualen Zahlendarstellungen getroffen werden (C o d e).

3.1. Verwendete Zahlendarstellung im Rechner

Das duale Zahlensystem ist die Grundlage aller intern im Rechner vorkommenden Zahlendarstellungen.

Vom Verwendungszweck, Zahlenbereich und von der Ein- und Ausgabe der Zahlen ist es abhängig, welche Art der Zahlendarstellung Verwendung findet.

3.1.1. Reine Dualzahlen

Das Rechnen mit reinen Dualzahlen gestattet einen relativ einfachen Aufbau des Rechenwerkes.

Im Rechner gelten allgemein die unter 2.2. beschriebenen Bedingungen und Rechenregeln. Bei der Addition und Subtraktion von reinen Dualzahlen müßte aber im Rechner zunächst das Vorzeichen der zu verknüpfenden Zahlen geprüft und danach das Rechenwerk entsprechend eingestellt werden.

Beispiel: a + b

Grundsätzlich würde das Rechenwerk zunächst auf Addition gestellt werden. Ist aber z. B. b eine negative Zahl, so müßte nachträglich das Rechenwerk auf Subtraktion geschaltet werden. Diese nochmalige Umschaltung des Rechenwerkes wird vermieden, indem negative Zahlen im Komplement dargestellt werden.

Zwei Varianten der Komplementbildung sind möglich:

<u>B-Komplement (komplementärer Code)</u>

Bei der Bildung des B-Komplements werden alle Ziffern der Dualzahl negiert (alle Ziffern 1 durch 0 ersetzt und umgekehrt); anschließend wird in der niedrigsten Stelle eine 1 addiert.

Beispiel: B-Komplement von $1\,0\,1\,1\,1\,0\,0\,0_2$

$$\overline{10111000} = 0\,1\,0\,0\,0\,1\,1\,1$$
$$+\quad 1$$
$$0\,1\,0\,0\,1\,0\,0\,0_2$$

Schneller wird das B-Komplement nach folgender Regel gebildet:

Die 1 mit der niedrigsten Stellenwertigkeit wird übernommen, alle anderen Bits mit höherer Wertigkeit werden umgekehrt.

Beispiel: 10111000_2
B-Komplement 01001000_2

(B-1) - Komplement (Inverser Code)

Bei der Bildung des (B-1) - Komplementes wird die gesamte Dualzahl negiert.

Beispiel: 1 1 1 0 0 1 0
(B-1) - Komplement $\overline{1\,1\,1\,0\,0\,1\,0} = 0\,0\,0\,1\,1\,0\,1$

Im D 4a wird mit dem B-Komplement gearbeitet.
An einem Beispiel soll die Addition einer negativen Zahl mit Hilfe des B-Komplementes erläutert werden.

Beispiel:

$$
\begin{array}{rr|rl}
 & +27_{10} & +11011_2 & \text{(1. Summand)} \\
+ & (-14_{10}) & +(-01110_2) & \text{(2. Summand)} \\
\hline
 & +13_{10} & 11011_2 & \\
 & & = +10010_2 & \text{-> B-Komplement} \\
\hline
 & & 1/01101_2 & = 13_{10}
\end{array}
$$

Der 1. Summand bleibt erhalten, vom 2. Summanden wird das B-Komplement gebildet. Nach erfolgter Addition muß das Ergebnis durch Streichen der Stelle mit der höchsten Stellenwertigkeit korrigiert werden.
Bei der Subtraktion im inversen Code muß am Schluß noch eine Korrektur des Ergebnisses durch Addition mit einer 1_2 erfolgen, sonst gilt der gleiche Rechenvorgang.

Bei einem Digitalrechner mit konstanter Wortlänge wird zur Darstellung des Vorzeichens in der Regel das Bit mit der höchsten Stellenwertigkeit verwendet.

Es gilt:

Vorzeichenbit mit 0 besetzt = positives Vorzeichen
Vorzeichenbit mit 1 besetzt = negatives Vorzeichen.

Die feste Wortlänge legt auch den zulässigen Zahlenbereich des Rechners fest.

Beispiel: Die Wortlänge soll 6 Bit betragen.

Der Aufbau des Wortes könnte folgendes Aussehen haben:

Z_0	Z_1	Z_2	Z_3	Z_4	Z_5
VZ	2^4	2^3	2^2	2^1	2^0

Stellenwertigkeit: VZ (Vorzeichenbit)

Die größte positive Zahl ist dann $+ 31_{10}$

$[0 \mid 1\,1\,1\,1\,1] \mathrel{\hat=} +1\,1\,1\,1\,1_2 = 31_{10}$

Die größte negative Zahl wäre $- 32_{10}$
(im Rechner im B-Komplement dargestellt)

$[1 \mid 0\,0\,0\,0\,0] \mathrel{\hat=} -1\,0\,0\,0\,0\,0_2 = -32_{10}$

Die Verwendung des Bits mit der höchsten Stellenwertigkeit als Vorzeichenstelle ist besonders günstig, weil bei der Bildung des Komplements dieses Bit automatisch mit dem richtigen Vorzeichen besetzt wird.

Wenn der Zahlenbereich des Rechners durch Rechenoperationen überschritten wird, tritt ein sogenannter Überlauf ein. Das führt zu Fehlern und der Rechner stoppt.

Sind im Maschinenwort die beiden ersten Bits mit Z_0 (Vorzeichenstelle) und Z_1 (Bit mit höchster Stellenwertigkeit) bezeichnet, so gilt für den Überlauf die folgende Regel:

Z_0	Z_1	. . . . .	Z_n

Ein Überlauf liegt vor, wenn bei Rechenoperationen die beiden höchsten Stellen Anlaß zu unterschiedlichen Über-

trägen geben:

d. h. 1.) Es erfolgt Übertrag von Z_1 nach Z_o ohne Übertrag von Z_o nach 1 Stelle vor Z_o (gedachte Stelle).

oder 2.) Es erfolgt Übertrag von Z_o nach 1 Stelle vor Z_o ohne Übertrag von Z_1 nach Z_o.

Für die nachfolgenden Beispiele wird wieder ein Wort mit einer Wortlänge von 6 Bit verwendet; der dezimale Zahlenbereich ist festgelegt mit $-32 \leq Z \leq 31$.

Beispiel:

Zu 1.)

```
    0. 1 1 0 0 0     + 24₁₀
  + 0. 1 0 0 0 0     + 16₁₀
  --------------     ------
    1. 0 1 0 0 0     + 40₁₀  > 31 → Überlauf
```

Zu 2.)

```
    1 0 1 0 0 0      - 24₁₀
  + 1 0 1 0 0 0    + (- 24₁₀)
  -------------    ---------
  1|0 1 0 0 0 0      - 48₁₀  < -32₁₀ → Überlauf
```

Zeichenerklärung:

◡ = Übertrag

◡ (gestrichelt) = kein Übertrag

Kein Überlauf erfolgt z. B. bei

```
    1 1 1 0 0 0       - 8
  + 1 1 1 0 0 0    + (- 8)
  -------------    -------
  1/1 1 0 0 0 0      - 16
  =============    =======
```

3.1.2. Tetradenverschlüsselung

Die Darstellung von Dezimalzahlen im Rechner mittels der Tetradenverschlüsselung ist sehr einfach und gestattet auch eine relativ einfache Ein- und Ausgabe. Sie bietet aber nicht die Vorteile der bekannten Stellenwertsysteme bezüglich der Durchführung der Grundrechenarten.

Dagegen ist das Rechenwerk einer Maschine, die als internen Code die Tetradenverschlüsselung verwendet, kompliziert und teuer.

Bei diesem Code wird stets jede einzelne Dezimalziffer durch eine vierstellige Dualzahl dargestellt.

Zuordnung zwischen Tetrade und Dezimalziffer

Tetrade	Dezimalziffer	
0 0 0 0	0	
0 0 0 1	1	
0 0 1 0	2	
0 0 1 1	3	
0 1 0 0	4	
0 1 0 1	5	
0 1 1 0	6	
0 1 1 1	7	
1 0 0 0	8	
1 0 0 1	9	
1 0 1 0	10	Pseudotetraden
1 0 1 1	11	Pseudotetraden
1 1 0 0	12	Pseudotetraden
1 1 0 1	13	Pseudotetraden
1 1 1 0	14	Pseudotetraden
1 1 1 1	15	Pseudotetraden

Von den 16 möglichen Kombinationen der Tetrade werden nur die ersten 10 zur Darstellung der Dezimalziffern 0 - 9 benötigt, die restlichen möglichen Kombinationen werden als Pseudotetraden bezeichnet.

Die Dezimalzahl 127,5 lautet in der Tetradenverschlüsselung

0001 0010 0111 , 0101

3.1.3. Halblogarithmische Zahlendarstellung

Für die Darstellung großer Zahlen mit beweglichem Komma (Gleitkomma) ist die Verwendung einer halblogarithmischen Schreibweise vorteilhaft. Dabei wird die Zahl als Produkt von Mantisse m und Basispotenz mit dem Exponenten e geschrieben

$$z = m \cdot B^{e}$$

Die Dezimalzahl 326,75 könnte also lauten:

$$3{,}2675 \cdot 10^{2} = 326{,}75 \cdot 10^{0} = 0{,}032675 \cdot 10^{4} \text{ usw.}$$

Um für den Rechner aber eine eindeutige Darstellung zu erhalten, wird meist eine Normalisierung der Mantisse gefordert.

Für die Normalisierung gelten folgende Regeln:

1. Die Mantisse muß dem Betrage nach kleiner als 1 sein
2. Die erste Ziffer nach dem Komma muß verschieden von Null sein.

Die Dezimalzahl 326,75 muß dann mit normalisierter Mantisse also folgende Darstellung erhalten:

$$= 0{,}32675 \cdot 10^{3}$$

Negative Gleitkommazahlen werden im Rechner nicht im Komplement dargestellt, sondern im direkten Code (Zahl u. Vorzeichen).

In der Interndarstellung wird meist (auch beim D 4a) als Basis die 2 gewählt.

Für die Umwandlung von Dezimalzahlen in Dualzahlen in

Gleitkommadarstellung wird folgendes Umrechnungsverfahren verwendet:

Die Dezimalzahl wird zunächst mit den bekannten Verfahren in eine normale Dualzahl umgewandelt. Anschließend wird das Komma so oft verschoben, bis es unmittelbar vor der ersten 1 steht. Die Anzahl der Verschiebungen gibt den Exponenten zur Basis 2 an (Linksverschiebung = Division der Dualzahl durch 2, Rechtsverschiebung = Multiplikation der Dualzahl mit 2).

Beispiel:

$315{,}34375_{10} = 100\ 111\ 011{,}\ 010\ 11_2$

(siehe Abschnitt 2.4.2.)

Dualzahl in Gleitkommadarstellung:

$$0{,}100\ 111\ 011\ 010\ 11 \cdot \underbrace{10}_{\hat{=}\ 2_{10}}{}^{1001\}\ \hat{=}\ 9_{10}}$$

Dabei müssen Basis und Exponent auch in Dualschreibweise dargestellt werden.

Beispiel:

$0{,}125_{10} = 0{,}001_2$

Dualzahl in Gleitkommadarstellung:

$0{,}1 \cdot 10^{-10}$ (dezimal: $0{,}5 \cdot 2^{-2}$)

Die Umrechnung der dualen Gleitkommazahl in eine Dezimalzahl erfolgt umgekehrt.
Zunächst wird der Exponent durch die entsprechende Anzahl von Kommaverschiebungen beseitigt und anschließend die entstandene Dualzahl nach den in Abschnitt 2.4. beschriebenen Regeln in eine Dezimalzahl umgewandelt.

4. Allgemeine Grundlagen der Programmierung

4.1. Einführung

4.1.1. Begriff und Aufgaben der Programmierung

Das Programm ist eine Folge von elementaren Anweisungen, nach denen der Rechner eine bestimmte Aufgabe löst. Den zur Lösung einer Aufgabe notwendigen Problemfunktionen werden bestimmte Maschinenfunktionen zugeordnet. Die Problemoperatoren müssen in Maschinenoperatoren transformiert werden. Die elementare Form eines Maschinenoperators wird als Befehl bezeichnet. Ein Programm besteht folglich aus einer geordneten Menge von Befehlen. Werden diese Befehle numeriert, so können durch eine Vorschrift, die im Rechner vorgegeben ist, die Befehle einzeln aufgerufen und abgearbeitet werden. Die Abarbeitung erfolgt in der Regel in der natürlichen Reihenfolge der Numerierung. Es ist jedoch möglich, diese natürliche Reihenfolge der Abarbeitung zu verlassen.

Die einzelnen Befehle, die ein Rechner ausführen kann, werden im Befehlsschlüssel oder in der Befehlsliste erläutert.

4.1.2. Algorithmen

"Ein Algorithmus ist eine genaue Vorschrift, nach der ein gewisses System von Operationen in einer bestimmten Reihenfolge auszuführen ist und nach der man alle Aufgaben eines gegebenen Typs lösen kann."

Die einfachsten Algorithmen sind die Regeln für die vier Grundrechenarten des Dezimalsystems.

Ein Algorithmus ist vor allem durch zwei Eigenschaften charakterisiert:

1. Die zwangsläufige Bestimmtheit (Determiniertheit) eines Algorithmus:

   Ein Algorithmus ist im Sinne der Logik eindeutig be-

grifflich bestimmt und abgegrenzt. Er kann in Form eines Rechenverfahrens durch endlich viele Anweisungen einem Rechner übergeben werden. Dabei hängt die Rechnung nicht vom Rechner und den benutzten Hilfsmitteln ab. Es handelt sich um einen Prozeß, der zum gleichen Resultat auch zu anderer Zeit mit anderen Hilfsmitteln führt.

2. Die Allgemeinverwendbarkeit des Algorithmus:
Ein Algorithmus dient zur Lösung aller Aufgaben gleichen Typs.
Der Algorithmus liefert für verschiedene Ausgangswerte (Parameter) in jedem Falle das entsprechende Resultat, wenn es sich um Aufgaben gleichen Typs handelt.

4.1.3. Arbeitsablauf bei der Programmierung

Folgende Arbeitsgänge müssen bei der Bearbeitung einer Aufgabe mit einem Rechner durchgeführt werden:

1. Formulierung und Analyse des Problems:
Eine häufig erforderliche Abstraktion führt zur speziellen, im allgemeinen bereits eingeschränkten Fragestellung.

2. Mathematische Formulierung der Fragestellung:
Die Aufgabe wird in eine mathematische Form gebracht. Im Ergebnis dieses Arbeitsprozesses liegt ein mathematisches Modell vor. Das erhaltene Modell kann von bestimmten Kriterien abhängig sein: Zeitberücksichtigung, Zufallsberücksichtigung, Eingehen von Erfahrungswerten, berücksichtigte Hypothesen.

3. Auswahl eines Lösungsverfahrens:
Das am besten geeignete Verfahren der numerischen Mathematik wird für die Lösung zugrunde gelegt. Mitunter erweist es sich als erforderlich, ein bekanntes Verfahren zu erweitern oder zu modifizieren oder ein neues Verfahren aufzubauen.Bei der Auswahl sollte bereits der zu benutzende Automatentyp berücksichtigt werden.

4. Erfassung eines Berechnungsablaufes
   Die geeignetste Form der Erfassung des Ablaufes der Berechnungen geschieht mittels eines Programmablaufplanes (siehe 4.2.).
5. Eigentliche Programmierung:
   a) Aufstellung des Pseudoprogrammes: Es handelt sich um ein Programm, in dem noch keine feste Speicherplatzverteilung vorgenommen wurde
   b) Organisation einer zweckmäßigen Speicherplatzverteilung
   c) Aufstellung des endgültigen Maschinenprogrammes mit fester Speicherplatzverteilung
   d) Herstellung des Eingabemediums (Lochkarte, Lochstreifen)
   e) Prüfen des Eingabemediums und Berichtigung von Fehlern
   f) Probelauf des Programmes
6. Ausführung der eigentlichen Berechnungen auf dem Rechner.
   Nachdem das Programm hergestellt und getestet wurde, können für die geforderten Varianten und Parameter die Aufgaben beliebig oft auf dem Rechner mit diesem Programm gelöst werden.

4.2. Programmablaufpläne

Der Programmablaufplan ist eine detaillierte, graphische Darstellung eines vorgegebenen Algorithmus und stellt eine Verbindung zwischen den mathematischen Verfahren und den Maschinenfunktionen dar.

Er enthält das arithmetische, organisatorische und logische Konzept der Lösung einer Aufgabe, d. h., er stellt den Ablauf des Lösungsweges graphisch dar.

Folgende Forderungen werden an einen Programmablaufplan gestellt:

a) Vollständige Beschreibung des mathematischen Verfahrens
b) Angabe der Eingangsgrößen, Zwischenwerte, Anfangs- und Endzustände

c) Möglichkeit zur Einführung von Teilprogrammen
d) Aufwand zur Darstellung soll gering sein
e) Der Programmablaufplan muß einen Gesamtüberblick gestatten.

Zwei Arten der Darstellung sind nach der TGL 22451 "Informationsverarbeitung - Datenfluß- und Programmablaufpläne - Sinnbilder" möglich, nämlich die

Kästchenmethode und die
Programmlinienmethode.

Die Programmlinienmethode stellt gegenüber der Kästchenmethode eine Einsparung an Arbeitsaufwand dar, deshalb wird im folgenden die Programmlinienmethode erläutert.

Programmlinienmethode:

Die Programmlinienmethode arbeitet mit einer den Programmablauf andeutenden Linie, die durch Symbole oder Anweisungen in Worten ergänzt wird.
Als Auszug aus der TGL 22451 werden auf den nächsten Seiten die Sinnbilder und ihre Bedeutung für den Programmablaufplan erläutert.

3. Vereinfachte Darstellung von Programmabläufen (Programmlinienmethode)

	Sinnbild	Bedeut. für Programmablaufpläne	Erläuterungen
3.1.		Zuordnung von Operationen (Anweisungen) zur Programmlinie	Einzelne oder zusammengefaßte Befehle in mathematischer oder textlicher Form; grundsätzlich rechts neben die vertikalen Programmlinien, jedoch nicht an die linken Verzweigungen zu schreiben; z.B. Ein- und Ausgaben, arithmetische Operationen.
3.2.	P{x > R} P{x = 0}	Verzweigen der Programmlinie nach rechts oder links Zusammenführen der Programmlinien von rechts oder links	Sprung im Ergebnis einer Vergleichsoperation. Die Vergleichsanweisung kann als Frage, z.B. x = 0 ?, oder mit Hilfe geschweifter Klammern geschrieben werden, z.B. P{x = 0}
3.3.	Blockende ?	Zusammenführen der Programmlinien von links oder rechts Verzweigen der Programmlinie nach links oder rechts	Sprung im Zyklus als Ergebnis einer Vergleichsoperation. Die Vergleichsanweisung kann als Frage, z.B. x = 0 ?, oder mit Hilfe geschweifter Klammern geschrieben werden, z.B. P{x = 0}
3.4.		Zusammenführen mehrerer Programmlinien von links und rechts	Die vertikalen Parallellinien müssen beim Verzweigen und Zusammenführen von rechts für eventuelle Beschriftungen ausreichenden Abstand voneinander haben; andernfalls sind Konnektoren zu verwenden.
3.5.		Kreuzung von abzweigenden und zusammenführenden Programmlinien (ohne verbindende Bedeutung)	Das Überschneiden von Programmlinien darf die Übersichtlichkeit und die Beschriftung dieser Linien nicht beeinträchtigen; andernfalls sind Konnektoren zu verwenden. Die Programmhauptlinie darf nicht gekreuzt werden.

Fortsetzung der Tabelle Seite 11

	Sinnbild	Bedeut. für Programmablaufpläne	Erläuterungen
3.6.	A, 4, A, 10, 4, 10	Zeichen-Konnektor (für Unterbrechung und Fortsetzung der Programmlinie aus zeichnerischen Gründen Mit Pfeilspitze bei Ab- und Einsprüngen nach und von links und rechts)	Zusammengehörige Konnektoren (Ab- und Einsprungstellen) müssen die gleiche Bezeichnung z.B. A, 4, 10 erhalten. Es können mehrere Absprungstellen zu einer Einsprungstelle führen.
3.7.	von Seite 2, 41, 41, nach Seite 3	Seiten-Konnektor (für Unterbrechung und Fortsetzung der Programmlinie auf einer anderen Seite)	z.B. Fortsetzung bei Konnektor 41 auf Seite 3
3.8.	i = 1... 3, A_i, A_1, A_2, A_3	Variabler Konnektor (für mehrfaches Verzweigen der Programmlinie entsprechend einer vorher festgelegten Bedingung)	Die Einsprungstelle richtet sich nach dem aktuellen Wert des Parameters, z.B. A_i = A_1 oder A_2 oder A_3
3.9.	x = 0 ?, n, j, P{x=0}, P, $\overline{P}$	Kennzeichnen der Wege nach Verzweigungen	Das Kennzeichnen der Wege in Abhängigkeit von Vergleichsergebnissen muß der Form der Vergleichsanweisung entspr. z.B. nach P{x=0} stehen P u. $\overline{P}$, nach: Blockende ? j (ja) und n (nein), nach Schalter A ? 1, 2
3.10.	1, a ⇒ x, a + b ⇒ c, i + 1 ⇒ i, 2, 1, x ⇒ c, 0 ⇒ i, 2	Korrekturen oder Ergänzungen	Ungültige Programmteile sind wegzustreichen. Neues oder zusätzliches Programmteil ist mit Hilfe von Konnektoren anzuschließen - wie im Beispiel angegeben.

Fortsetzung der Tabelle Seite 12

	Sinnbild	Bedeut. für Programmablaufpläne	Erläuterungen
3.11.		Unterprogramm - Block, Programmteil	Der Name des Unterprogrammes, Programmteil ist rechts neben die Figur zu schreiben. Mit Sinnbild Nr. 3.14 sind die Eingangs- u. Resultatswerte anzugeben.
3.12.	B E	Beginn und Ende eines Unterprogramms, Programmteiles	Für die Detaillierung von Unterprogrammen oder Programmteilen außerhalb der Programmhauptlinie.
3.13.	START HALT STOP	Programmanfang Programmende	Es sind START, STOP oder HALT - HALT bei Zwischenhalt - in das Sinnbild einzuschreiben oder STOP und Zwischenhalt als Anweisung an die Programmlinie zu schreiben.
3.14.	START STOP	Parallellinie (Zuordnen von Bemerkungen u. Parameterangaben zur Programmlinie)	z.B. Variable, Zahlenbereiche, Anfangs- und Resultatwerte, jedoch nicht Ein- und Ausgaben.

4. Kennzeichen für Operationsbeschreibungen

	Kennzeichen	Bedeutung	Erläuterungen
4.1.	⇐ oder :=	Ergibt	Pfeilspitze oder Doppelpunkt weist auf Ergebnis. Auch rechts gerichtet statthaft. z.B. $i \Leftarrow i+1$
4.2.	⟶	Transport	z.B. $a \rightarrow b$
4.3.	⟨ ⟩	Inhalt (einer Speicherzelle)	z.B. $\langle B \rangle = a$
4.4.	⟪ ⟫	Substituierter Inhalt (einer Speicherzelle)	z.B. $\langle\langle C \rangle\rangle = D$
4.5.	⟩ ⟨	Adresse (einer Speicherzelle)	z.B. $\rangle a \langle = B$

4.2.1. Unverzweigte Geradeausprogramme

Bei unverzweigten Geradeausprogrammen wird ohne Entscheidungen (Test) Anweisung nach Anweisung abgearbeitet.

Beispiel: $e = \frac{a - b}{c + d}$ gegeben: a, b, c, d

START

a, b, c, d

Z := a - b

N := c + d

e := Z/N

e

STOP

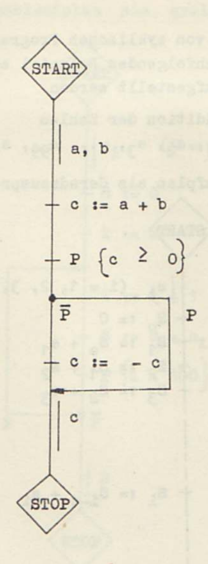
4.2.2. Verzweigte Geradeausprogramme

Bei Geradeausprogrammen mit Verzweigung erfolgt durch Tests (Abfrage und Entscheidung) eine Verzweigung des Programms, jedoch keine Rückführung zu einer Anweisung, die schon einmal abgearbeitet wurde.

Beispiel: $|a + b| = c$ gegeben: a, b

Es gilt: $|a + b| = a + b$ für $a + b \geq 0$

$|a + b| = -(a + b)$ für $a + b < 0$



4.2.3. Zyklische Programme

Ein Zyklus (eine Schleife) entsteht, wenn ein Teil des Programmablaufes mehrfach durchlaufen wird.

Man erkennt zyklische Programme an den nach oben führenden Linien. Auf diesen Linien dürfen keine Anweisungen stehen.

Der Zyklus wird immer nach einem Test abgebrochen.

Man unterscheidet zwischen Induktions- und Iterationszyklen.

4.2.3.1. Induktionszyklen

Bei Induktionszyklen wird die Anzahl der Durchläufe im Zyklus durch Zählen und Vergleich mit der vorgegebenen Anzahl von Durchläufen bestimmt.

Um die Vorteile von zyklischen Programmen besser zu erkennen, soll nachfolgendes Beispiel zunächst als Geradeausprogramm aufgestellt werden.

Beispiel: Addition der Zahlen
$a_1, a_2, a_3, \ldots, a_{99}, a_{100}$

Programmablaufplan als Geradeausprogramm:

START

$a_i \quad (i = 1, 2, 3, \ldots, 100)$

$S_o := 0$

$S_1 := S_o + a_1$

$S_2 := S_1 + a_2$

$S_3 := S_2 + a_3$

$S_i := S_{i-1} + a_i$

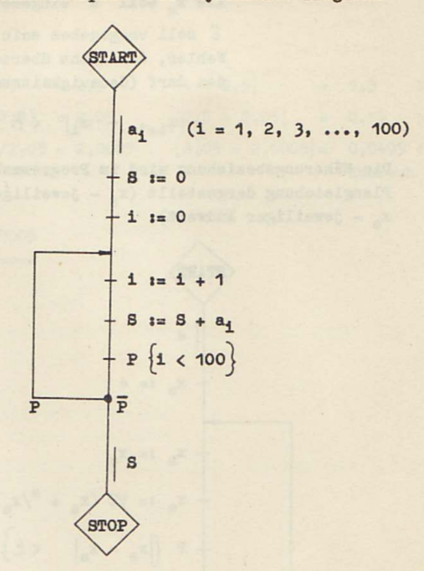
$S_{100} := S_{99} + a_{100}$

S_{100}

STOP

Diese Darstellung ist sehr aufwendig für Programmierer und Automaten, da eine große Anzahl von Operationen vorkommt. Für eine mehrfach vorkommende Operationsanweisung wird in zyklischen Programmen nur eine benutzt und die Anzahl der Durchführungen gezählt. Als Zählgröße wird i, als Zwischen- und Endsumme S verwendet.

Programmablaufplan als zyklisches Programm



Für S und i werden zunächst die Anfangsbedingungen geschaffen. Zur anfänglichen Summe S = 0 wird a_i innerhalb des Zyklus so oft addiert, bis i = 100 ist, d. h. wenn die letzte Zahl a_{100} addiert ist, wird der Zyklus abgebrochen.

4.2.3.2. Iterationszyklen

Bei Iterationszyklen hängt die Anzahl der Durchläufe im Zyklus von der geforderten Genauigkeit ab.

Beispiel: Berechnung von $x = \sqrt{a}$ nach dem Näherungsverfahren:

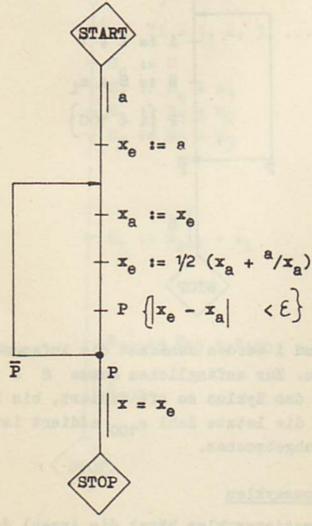
$$x_{i+1} = 1/2 \left(x_i + \frac{a}{x_i}\right) \qquad i = 0, 1, 2, 3 \ldots$$

Als x_o soll a eingesetzt werden.

ε soll vorgegeben sein als absoluter Fehler, der nicht überschritten werden darf (Genauigkeitsschranke).

$$|x_{i+1} - x_i| < \varepsilon$$

Die Näherungsbeziehung wird im Programmablaufplan als Plangleichung dargestellt (x_a - jeweiliger Anfangswert, x_e - jeweiliger Endwert).



Zahlenbeispiel: $x = \sqrt{4}$ $\varepsilon = 0{,}05$

$x_{i+1} = 1/2\ (x_i + {}^a/x_i)$; bei $(x_{i+1} - x_i) < \varepsilon$ -> Abbruch

$x_o = 4$

$x_1 = 1/2\ (4 + {}^4/4) = 2,5 \qquad |4 - 2,5| = 1,5 > \varepsilon$

$x_2 = 1/2\ (2,5 + {}^4/2,5) = 2,05 \qquad |2,5 - 2,05| = 0,45 > \varepsilon$

$x_3 = 1/2\ (2,05 + {}^4/2,05 = 2,0005 \qquad |2,05 - 2,0005| = 0,0495 < \varepsilon$

Abbruch

$x = x_3 = 2,0005$

====================

Literaturverzeichnis

(1)	Kolb	Programmieranleitung für den Digitalrechner D 4a - 1965 (nicht veröffentlicht)
(2)	Müller/Spreitz	Aufbau, Wirkungsweise und Programmierung des digitalen Rechenautomaten D 4a (veröffentlichtes Manuskript aus dem IDV Dresden)
(3)	Kämmerer, W.	Ziffernrechenautomaten Berlin, Akademie - Verlag 1960
(4)	Bachmann, K.-H.	Programmierung für Digitalrechner VEB Deutscher Verlag der Wissenschaften - Berlin 1962
(5)	Trachtenbrot, B.A.	Wieso können Automaten rechnen ? VEB Deutscher Verlag der Wissenschaften - Berlin 1959
(6)		Manuskript des Entwurfs für ein Einführungsheft vom VEB Rechenelektronik Meiningen



10.4. VEB Möbelw

9 Uhr

1.